Modelo gravitacional para clasificación

Alberto Cano, José María Luna, Amelia Zafra, Sebastián Ventura

Resumen— **La gravedad es una interacción fundamental, cuyo concepto y efectos han sido aplicados a la clasificación de datos. El principio de la clasificación gravitatoria es clasificar los ejemplos mediante la comparación de la atracción gravitatoria a las distintas clases. Este artículo presenta un algoritmo de clasificación gravitatoria que mejora los modelos gravitatorios existentes y soluciona algunos de sus problemas. El algoritmo propuesto, denominado IGCA (Improved Gravitation Classification Algorithm), es capaz de trabajar con atributos numéricos y nominales, filtrar atributos ruidosos o irrelevantes, y alcanzar buenos resultados sobre conjuntos de datos no balanceados. La propuesta emplea una matriz de pesos para describir la importancia de cada atributo en la clasificación de cada clase y mejora la exactitud en la clasificación al considerar tanto la información global como local de los datos, especialmente en los ejemplos cercanos a la frontera entre clases. La propuesta se evalúa y compara con otros 6 algoritmos de clasificación, incluyendo un modelo gravitatorio previo, sobre 30 conjuntos de datos. Los resultados obtenidos de la experimentación muestran los buenos resultados de nuestro modelo gravitatorio y se validan mediante tests estadísticos no paramétricos.**

Palabras clave— **Aprendizaje automático, clasificación gravitatoria, GPUs, estrategias evolutivas, CMA-ES**

I. Introducción

La tarea de clasificación consiste en la predicción de la clase para un ejemplo dado por medio de los valores de sus atributos y es una de las tareas fundamentales en aprendizaje automático supervisado. Existen muchos paradigmas y técnicas que solucionan la tarea de clasificación, tales como las redes neuronales [1], las máquinas de vector soporte [2], y los métodos basados en las instancias [3].

El algoritmo del vecino más cercano (NN) [4] es un método basado en las instancias cuyo principio de clasificación consiste en asignar la clase a un ejemplo desconocido con la clase del ejemplo conocido más cercano. La versión extendida de este método a k vecinos y sus derivados constituyen una de las familias de algoritmos más influyentes en clasificación y han demostrado su eficacia en muchos ámbitos [5]. Sin embargo, el principal problema de estos métodos es que se deterioran severamente al tratar con datos ruidosos o en conjuntos de datos de alta dimensionalidad, ya que su desempeño se vuelve muy lento y su exactitud tiende a decaer conforme en aumenta la dimensionalidad de los datos, especialmente cuando las clases no son separables o se encuentran solapadas [6].

En los últimos años, se han propuesto nuevos métodos basados en las instancias que emplean el principio de la clasificación de datos gravitatoria (DGC) para resolver estos problemas de los clasificadores de los vecinos más cercanos [7], [8], [9]. Los modelos DGC se inspiran en los principios de la ley universal de Newton para simular la atracción entre ejemplos. Estos métodos basados en gravedad extienden el concepto del vecino más cercano al dominio de la gravedad tal y como ocurre con los objetos del mundo real. El principio de la clasificación gravitatoria consiste en calcular la atracción gravitatoria entre un ejemplo y el resto de ejemplos, concluyendo que el grado de pertenencia de un objeto a una clase aumenta conforme la gravedad entre dicho ejemplo y el resto de ejemplos de dicha clase aumenta.

Este artículo presenta un modelo de clasificación gravitatoria mejorado (IGCA) que compara la atracción gravitatoria para las diferentes clases y predice la clase con mayor magnitud. La propuesta mejora los modelos de clasificación gravitatoria previos mediante el aprendizaje de los pesos de los atributos para cada clase, y soluciona algunos de sus problemas, tales como el manejo de los atributos nominales, la eficacia sobre conjuntos de datos no balanceados, y el filtrado de atributos ruidosos o irrelevantes. Los pesos de los atributos en la clasificación de cada clase se aprenden mediante el conocido algoritmo CMA-ES (covariance matrix adaptattion evolution strategy) [10]. La propuesta mejora la exactitud de los resultados considerando tanto la información global, como local de los datos, lo que ayuda en la predicción de la clase especialmente en los ejemplos cercanos a la frontera entre clases. El proceso de aprendizaje y la evaluación de la función objetivo de la población del algoritmo CMA-ES se acelera mediante el uso de GPUs, que ha demostrado ser una metodología efectiva en el aumento del rendimiento y el descenso del tiempo de ejecución [11].

Los experimentos se han llevado a cabo sobre 30 conjuntos de datos con diferentes características para evaluar los resultados de la propuesta y compararla con otros 6 algoritmos de clasificación, seleccionados de la herramienta software KEEL [12] y su repositorio de datos [13]. Los algoritmos comparados incluyen algunas de las técnicas de clasificación más relevantes presentadas hasta la fecha. Los resultados obtenidos muestran el buen desempeño de nuestra propuesta, obteniendo resultados significativamente mejores en términos de exactitud. El estudio experimental incluye un análisis estadístico basado en los tests no paramétricos [14], [15] de Bonferroni–Dunn [16] y Wilcoxon [17], con el objetivo de evaluar si existen diferencias significativas entre los resultados de los algoritmos.

Departamento de Informática y Análisis Numérico. E-mail: {i52caroa,i32luarj,azafra,sventura}@uco.es .

El artículo introduce a continuación una sección donde se presentan algunas definiciones y se revisa el trabajo relacionado con modelos de clasificación basados en gravedad. La sección III describe el algoritmo propuesto. La sección IV presenta el estudio experimental, cuyos resultados se muestran en la sección V. Finalmente, la sección VI presenta las conclusiones del trabajo.

II. Clasificación Gravitatoria

Esta sección introduce la ley de gravitación universal de Newton, expone los trabajos relacionados de clasificación gravitatoria y sus principios básicos.

A. Ley de la Gravitación Universal de Newton

Newton publicó en 1687 la ley de gravitación universal, que enuncia que todo punto de masa en el universo atrae cualquier otro punto de masa con una fuerza que es directamente proporcional al producto de sus masas e inversamente proporcional al cuadrado de la distancia. Esta fuerza de atracción se calcula como:

$$\vec{F} = -G\frac{m_1 m_2}{r^2}\vec{u}_{12}$$

donde F es la fuerza entre los objetos, G representa la constante de gravitación universal, m_1 es la masa del primer objeto, m_2 es la masa del segundo objeto, y r es la distancia entre los objetos. La relatividad general generaliza la ley de Newton, proporcionando una descripción unificada de la gravedad como una propiedad geométrica del espacio–tiempo. La curvatura del espacio–tiempo se suele representar mediante diagramas de incrustación [18]. Estos diagramas muestran el campo gravitatorio que rodea a un objeto en el espacio. Cada punto del espacio tiene un valor del campo proporcional a la función:

$$\vec{g} = -G\frac{M}{r^2}\vec{u}_{12}$$

B. Modelos de clasificación basados en gravedad

En cuanto a los modelos de clasificación basados en gravedad, C. Wang y Y. Q. Chen [8] presentaron en 2005 una mejora del vecino más cercano (Nearest Neighbor, NN) utilizando el concepto de colapso gravitacional para reducir y definir los límites de la distribución de cada clase, ya que el rendimiento del clasificador NN se reduce significativamente con el aumento de la superposición de la distribución de diferentes clases. Su algoritmo genera prototipos para el clasificador NN por la migración de las instancias siguiendo la órbita gravitatoria.

Hay más contribuciones que aplican la teoría de la gravitación a clasificación, como la propuesta de B. Yang et al. [19] para la detección de intrusiones en la red o la de Y. Zong-Chang [9], que propuso en 2007 un modelo gravitatorio vectorial derivado del análisis geométrico del clasificador lineal.

El trabajo más reciente y completo en relación con la clasificación de datos mediante gravedad fue presentada por L. Peng et al. [7] en 2009. El modelo emplea un vector de pesos para describir la importancia de los atributos en el cálculo de las distancias entre los ejemplos. Los pesos se optimizan mediante un algoritmo iterativo y la gravedad de una partícula (conjunto de ejemplos) a las clases se calcula usando la masa de la partícula (número de instancias representadas) y la distancia a su centroide, que representa el centro de masas. El algoritmo crea las partículas empleando el principio de máxima distancia. Sin embargo, este método reduce la exactitud, especialmente en áreas alejadas del centroide de las partículas y a lo largo de la frontera entre clases.

C. Principios de la clasificación gravitatoria

Esta sección presenta los principales conceptos y principios de la clasificación gravitatoria, que consiste en asignar a una instancia la clase con la mayor magnitud del campo gravitatorio. Por lo tanto, dado un ejemplo $\bar{x}$ y un conjunto de datos con k clases, el algoritmo calcula la gravedad a $\bar{x}$ usando las instancias de entrenamiento para las diferentes clases, y finalmente predice la clase con la mayor magnitud.

$$Clase(\bar{x}) = max\left\{g(\bar{x}, c_1), g(\bar{x}, c_2), \ldots, g(\bar{x}, c_k)\right\}$$

donde $g(\bar{x}, c)$ es el campo gravitatorio en el punto donde se localiza el ejemplo $\bar{x}$, generado por las instancias de la clase c_i. Una diferencia significativa entre nuestra propuesta y los modelos de clasificación previos es precisamente cómo se calcula la gravedad considerando las distancias entre los ejemplos, cuya descripción se proporciona en la sección III-A.

La Figura 1 muestra una representación bidimensional del pozo gravitatorio para un conjunto de datos de tres clases. Los ejemplos de las distintas clases generan una atracción gravitatoria cuya magnitud se representa mediante un gradiente. Los modelos gravitatorios previos calculan la distancia al centroide de las partículas, que representa el centro de masas del cúmulo de instancias. En nuestro caso, nuestra propuesta emplea todas las instancias. La ventaja de usar todas las instancias para calcular el campo gravitatorio es que proporciona una buena generalización allí donde no hay instancias de entrenamiento cercanas (información global). Sin embargo, la debilidad del uso del centroide es la pérdida de información acerca de la forma de la partícula (información local), lo que no ocurre cuando se usan todas

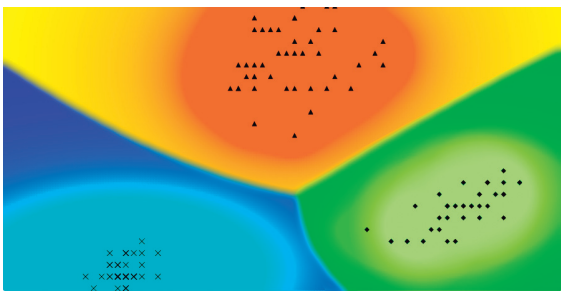
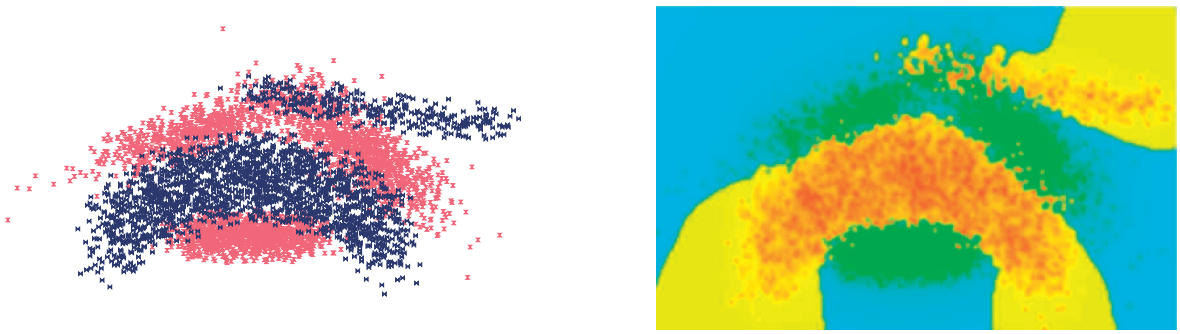


Fig. 1: Gradiente del campo gravitatorio



(a) Conjunto de instancias

(b) Predicción gravitatoria de la clase

Fig. 2: Conjunto de datos banana

las instancias. Este comportamiento es especialmente notorio cuando la forma del cúmulo de instancias es irregular, por lo que la frontera entre las clases no puede diferenciarse correctamente. La Figura 2 muestra el conjunto de datos banana (2 atributos, 2 clases) y la predicción de la clase de acuerdo a la magnitud del campo gravitatorio. La complejidad y el solapamiento de las clases convierte la clasificación de este conjunto de datos en un problema difícil de resolver.

III. Modelo gravitatorio

Esta sección presenta el algoritmo propuesto, cuyas principales características se muestran a continuación. Primero es necesario establecer la definición más apropiada de distancia para este problema. A continuación, se describe el procedimiento para calcular los pesos óptimos de los atributos para las distintas clases, y se presenta una discusión de cómo mejorar la eficacia del modelo sobre conjuntos de datos no balanceados, que era uno de los problemas de los modelos gravitatorios previos. Finalmente, se detallan los pasos del algoritmo para la construcción del clasificador.

A. Definición de distancia

La gravedad es inversamente proporcional al cuadrado de la distancia. La gravedad de Newton se formula en base a la geometría euclídea, que es la más natural. La distancia euclídea entre dos objetos $\bar{x}_1$ y $\bar{x}_2$ se define como:

$$d(\bar{x}_1, \bar{x}_2) = \sqrt{\sum_{i=1}^{f} (x_{1i} - x_{2i})^2}$$

donde el número de atributos (dimensiones) es f. Todos los atributos numéricos deben ser normalizados al intervalo $[0, 1]$ para un cálculo equitativo de la distancia. Con el objetivo de encontrar un criterio razonable para calcular la distancia entre las etiquetas de los atributos nominales se emplea la métrica de solapamiento que se define como:

$$\delta(x_{1i}, x_{2i}) = \begin{cases} 0 & \text{if} \quad x_{1i} = x_{2i} \\ 1 & \text{if} \quad x_{1i} \neq x_{2i} \end{cases}$$

Esta definición de distancia considera a todos los atributos igualmente relevantes, pero la selección de características ha demostrado mejorar los resultados de los algoritmos de clasificación [20], [21]. La selección de características permite la detección y filtrado de atributos irrelevantes, considerando la entropía de la distribución de las instancias conforme a los atributos y clases.

Peng et al. [7] demostraron el buen funcionamiento del ponderamiento del peso de los atributos en el cálculo de la distancia mediante un algoritmo iterativo. Sin embargo, el peso de cada atributo puede ser distinto para cada clase, puesto que la distribución de las instancias no es la misma para todas las clases. Por lo tanto, en vez de emplear un vector de pesos de atributos de longitud f, nuestro algoritmo emplea una matriz de pesos atributo–clase $W[f, k]$, donde k es el número de clases.

$$W = \begin{bmatrix} w_{1,1} & \dots & w_{1,k} \\ \dots & \dots & \dots \\ w_{f,1} & \dots & w_{f,k} \end{bmatrix}$$

Finalmente, la función de distancia se reescribe para tener en cuenta los pesos de los atributos para cada clase:

$$d(\bar{x}_1, \bar{x}_2, c) = \sqrt{\sum_{i=1}^{f} w_{i,c} \cdot (x_{1i} - x_{2i})^2}$$

B. Optimización de los pesos atributo–clase

La optimización de la matriz de pesos atributo–clase W es un problema de optimización real cuya dimensión, $f \cdot k$, depende del número de atributos y del número de clases. Uno de los algoritmos evolutivos más potentes en la resolución de problemas reales mono-objetivo no lineales y no convexos es CMA-ES [10], que es una estrategia evolutiva de adaptación de la matriz de covarianza. Las principales ventajas de CMA-ES residen en sus propiedades invariantes, conseguidas con un cuidadoso diseño de los operadores de selección y genéticos, y en su eficiente autoadaptación en la distribución de la mutación. CMA-ES no necesita una complicada parametrización puesto que la búsqueda de los mejores paráme-

tros es considerada parte del diseño del algoritmo, y no de su aplicación. Por lo tanto, nuestro modelo emplea el algoritmo CMA-ES para optimizar los valores reales de la matriz de pesos atributo–clase.

C. Eficacia con datos no balanceados

El problema del aprendizaje con datos no balanceados en un reto relativamente reciente que ha atraído la atención de la investigación [22]. El problema se debe a que la representación de los datos no es equitativa, existen grandes diferencias en la cantidad de datos que representan a cada una de las clases. Uno de los graves problemas extraídos de las conclusiones del artículo de Peng et al. [7] es el mal funcionamiento de su algoritmo gravitatorio en este tipo de conjuntos de datos no balanceados. Este problema sería aumentado en nuestro algoritmo, puesto que todas las instancias de entrenamiento participan en la decisión de la clase de una nueva instancia. La atracción gravitatoria de las instancias de las clases minoritarias sería eclipsada por aquellas de las clases mayoritarias. Por lo tanto, la gravedad de una instancia hacia una clase debe ser ponderada con respecto al número de instancias de esa clase y el número total de instancias. De esta forma, se consigue un balanceo y compromiso entre las fuerzas gravitatorias de las clases minoritarias y mayoritarias. Finalmente, la función de atracción gravitatoria de una instancia $\bar{x}$ a una clase c se define como:

$$g(\bar{x}, c) = (1 - \frac{n_c - 1}{n}) \cdot \sum_{i=1}^{n} \frac{1}{d(\bar{x}_i, \bar{x}, c)^2} \quad | \; \bar{x}_i \, \epsilon \, c$$

donde n_c es el número de instancias de la clase c y n es el número de instancias del conjunto de datos.

D. Algoritmo

Esta sección resume los pasos ejecutados por el algoritmo para la obtención del clasificador. Primero es necesario escalar todos los atributos y normalizarlos al intervalo [0,1]. El problema del cálculo absoluto de la distancia es que depende del rango y los valores del dominio de los atributos. Por ello, las distancias en atributos con dominios mayores diferirían de las de dominios menores, y en consecuencia, normalizarlos nos evita este problema relativizando las distancias. Para cada atributo, se define el valor máximo del dominio f_{max} y el mínimo f_{min}. Para cada instancia, se normalizan sus atributos siguiendo el esquema:

$$f_{norm} = \frac{f - f_{min}}{f_{max} - f_{min}}$$

A continuación, se deben encontrar los valores óptimos de los pesos de la matriz W atributo–clase con el objetivo de discriminar atributos ruidosos o irrelevantes para cada clase. Para ello, se codifica una población de genotipos reales para representar los valores de los pesos. Los valores iniciales de los pesos es 0.5, es decir, todos los atributos son inicialmente considerados igualmente relevantes para todas las clases. Se ejecuta el algoritmo CMA-ES para optimizar dichos pesos de acuerdo con la función de ajuste, que evalúa la exactitud del clasificador candidato usando los valores correspondientes de la matriz de pesos. Cuando se alcanza alguno de los criterios de parada, se toma la mejor solución como los pesos del clasificador final.

IV. Estudio experimental

Esta sección describe los detalles de los experimentos realizados en diferentes conjuntos de datos para evaluar las capacidades de la propuesta y compararla con otros métodos de clasificación. A continuación se presentan los conjuntos de datos empleados y los algoritmos utilizados, la descripción de la metodología experimental y las pruebas estadísticas utilizadas para la validación de los resultados.

A. Conjuntos de datos

Los conjuntos de datos utilizados en los experimentos han sido seleccionados de la página web de repositorio de KEEL [13] y son muy variados en su grado de complejidad, el número de clases, el número de atributos, y el número de instancias. El número de clases diferentes oscila de 2 a 11, el número de atributos entre dos y 60, y el número de instancias varía de 148 a 12.690. La Tabla I resume la información acerca de estos conjuntos de datos. Los conjuntos de datos se dividen en 10 particiones siguiendo el método de validación cruzada 10-fold cross validation [23], [24] y están disponibles para facilitar las comparaciones futuras con otros métodos.

B. Configuración de la experimentación

En esta sección se describe la configuración de la experimentación, los algoritmos empleados en la comparativa y sus parámetros, que se han obtenido de la herramienta KEEL [12]. En total, 6 algoritmos han sido seleccionados y comparados con nuestra propuesta con el objetivo de determinar si el funcionamiento del nuestro es competitivo en distintos conjuntos de datos. A continuación, se describen los distintos algoritmos empleados:

- DGC [7]: Algoritmo gravitatorio propuesto por Peng et al. que emplea pesos para describir la importancia de cada atributo en el modelo de clasificación gravitatorio. Emplea cúmulos o partículas para agrupar las instancias más cercanas, con el objetivo de reducir el número de instancias y la complejidad del conjunto de datos. La fuerza gravitatoria de una partícula a cada clase se calcula usando el centro de gravedad o centro de masas del cúmulo y el número de instancias representadas por dicho cúmulo.
- KNN [25]: El clásico clasificador de los k vecinos más cercanos clasifica una instancia con la clase del mayor número de sus vecinos. El criterio de vecindad se define como las k instancias con menor distancia

TABLA I: Complejidad de los conjuntos de datos

Dataset	#Instancias	#Atributos	#Clases
Balance	625	4	3
Banana	5300	2	2
Bupa	345	6	2
Car	1728	6	4
Dermatology	366	34	6
Ecoli	336	7	8
Flare	1066	11	6
German	1000	20	2
Glass	214	9	7
Haberman	306	3	2
Hayes-Roth	160	4	3
Heart	270	13	2
Hepatitis	155	19	2
Ionosphere	351	33	2
Iris	150	4	3
Lymphography	148	18	4
Monk-2	432	6	2
New-thyroid	215	5	3
Nursery	12690	8	5
Page-blocks	5472	10	5
Phoneme	5404	5	2
Pima	768	8	2
Sonar	208	60	2
Tae	151	5	3
Thyroid	7200	21	3
Tic-tac-toe	958	9	2
Vehicle	846	18	4
Vowel	990	13	11
Wine	178	13	3
Yeast	1484	8	10

a la instancia a clasificar. El número de vecinos seleccionado es 3 puesto que proporciona los mejores resultados.

- SFLSDE [26]: Este método basado en evolución diferencial genera instancias prototipo artificiales a partir de las verdaderas instancias de entrenamiento. Una vez ajustada la ubucicación de los prototipos, se evalúa mediante un clasificador del vecino más cercano.
- C-SVM [27]: Máquina de vector soporte que mapea los valores de entrada a un espacio de alta dimensionalidad a través de un kernel elegido a priori. En este espacio se construye una solución lineal con la propiedad que proporciona una buena generalización. El tipo de kernel seleccionado para la comparativa es el polinómico, puesto que proporciona mejores resultados que el kernel lineal o RBF.
- SMO [28]: Implementa el algoritmo de optimización mínimo secuencial de Platt's para entrenar a una máquina de vector soporte usando un kernel polinómico o RBF. Esta implementación reemplaza los valores omitidos y transforma los atributos nominales a binarios, y los normaliza. Los problemas multiclase se resuelven mediante clasificación por pares de clases. El tipo de kernel seleccionado para la comparativa es el polinómico, puesto que proporciona mejores resultados que el kernel RBF.
- NNEP [29]: Este método obtiene el diseño de una red neuronal y simultáneamente estima los pesos adecuados del modelo empleando un algoritmo de programación evolutiva. De esta forma, el modelo de red neuronal se obtiene de los datos de entrenamiento y se compara con los datos de test para evaluar su generalización.

Los autores de los modelos establecieron en sus respectivos trabajos diferentes configuraciones de los parámetros de sus métodos. Los parámetros que nosotros hemos empleado en el estudio comparativo son los parámetros óptimos encontrados por los autores en sus respectivos estudios experimentales.

Nuestra propuesta ha sido implementada en el software JCLEC [30] y sus únicos parámetros son los requeridos por el algoritmo CMA-ES. Precisamente, CMA-ES no necesita de una parametrización compleja, puesto que encontrar los valores óptimos se considera como parte del diseño del algoritmo, y no de su aplicación. Para la ejecución de CMA-ES, hemos empleado los valores por defecto y sólo es necesario detallar algunos de ellos a modo informativo. CMA-ES emplea un cromosoma cuya dimensionalidad en nuestro problema es (C_L), resultado del producto del número de clases y del número de atributos. Cada gen del cromosoma representa el peso de un atributo para una clase. Los valores iniciales del cromosoma se establecen a 0,5 y la desviación inicial a 0,3. El criterio de parada es el estancamiento de los valores de la función objetivo por debajo de $1E^{-13}$ y el número máximo de generaciones se sitúa en 500. El tamaño de población se adapta a la complejidad del problema y se calcula como $4 * \log {C_L}^2$. Para CMA-ES, el tamaño de población puede ser libremente elegido, porque previene de una convergencia prematura, incluso para poblaciones pequeñas. Valores de población bajos conducen habitualmente soluciones locales más rápidamente, mientras que poblaciones más grandes pueden conducir al óptimo global. El número de reinicios es dos y el tamaño de la población se mantiene constante.

Todos los experimentos se han repetido con 10 semillas diferentes para los métodos estocásticos, y los resultados medios son los que se muestran en las tablas. Todos los algoritmos se han ejecutado sobre los conjuntos de datos empleando validación cruzada 10-fold cross validation. Los experimentos se han ejecutado en un PC con un procesador Intel core i7 a 2.66 GHz, 12 GB DDR3 de memoria, dos gráficas NVIDIA GTX 480 para acelerar el cálculo de la función objetivo bajo el modelo de programación NVIDIA CUDA. El sistema operativo era GNU/Linux Ubuntu 11.04 64 bit.

C. Análisis estadístico

Para poder analizar los resultados de los experimentos es necesario realizar algunos tests estadísticos no paramétricos con el objetivo de validar los resultados y las conclusiones [14], [31]. Para evaluar si existen diferencias significativas en los resultados de los distintos algoritmos, primero se realiza el test de Iman y Davenport. Este test no paramétrico fue recomendado por Demsar [32], y se aplica para obtener el ranking de los K algoritmos sobre los N

TABLA II: Resultados de exactitud mediante 10-fold cross-validation

Data set	IGCA	DGC	KNN	SFLSDE	CSVM	SMO	NNEP
Balance	0.8966	0.8999	0.8337	0.8789	0.9168	0.8790	**0.9669**
Banana	**0.8952**	0.8931	0.8864	0.8789	0.5517	0.5517	0.7411
Bupa	0.6744	0.6527	0.6066	0.6225	0.7014	0.5789	**0.7240**
Car	**0.9523**	0.9126	0.9231	0.8474	0.8519	0.9385	0.9126
Contraceptive	0.4945	0.4954	0.4495	0.4476	0.5126	0.5116	**0.5318**
Dermatology	0.9544	0.9170	0.9690	0.9541	0.9635	**0.9718**	0.9403
Ecoli	**0.8217**	0.7672	0.8067	0.7921	0.7592	0.7709	0.7324
German	0.7322	0.7020	0.6960	0.7157	0.7360	**0.7450**	0.7313
Glass	**0.7036**	0.6893	0.7011	0.6679	0.6259	0.5924	0.6319
Haberman	0.7171	0.7277	0.7058	0.7100	0.7287	0.7353	**0.7489**
Hayes-Roth	**0.8400**	0.7738	0.2500	0.6479	0.6062	0.5271	0.6792
Heart	**0.8452**	0.8119	0.7741	0.8136	0.8444	0.8444	0.8173
Hepatitis	0.8628	0.8343	0.8251	0.8514	0.8356	**0.8900**	0.8431
Ionosphere	**0.9311**	0.6724	0.8518	0.8765	0.8803	0.8889	0.9165
Iris	0.9533	0.9533	0.9400	0.9444	**0.9667**	0.9600	0.9444
Lymphography	0.8140	0.8033	0.7739	0.7725	0.8398	**0.8477**	0.7693
Monk-2	**0.9995**	0.9982	0.9629	0.8910	0.8061	0.8061	0.9931
New-thyroid	0.9786	0.8684	0.9537	0.9663	**0.9816**	0.9024	0.9725
Nursery	**0.9696**	0.9378	0.9254	0.7084	0.7407	0.9313	0.9037
Page-blocks	0.9508	0.9268	0.9591	0.9476	**0.9671**	0.9270	0.9476
Phoneme	0.8718	0.8471	**0.8849**	0.8164	0.7733	0.7734	0.7903
Pima	0.7451	0.6662	0.7319	0.7388	**0.7710**	**0.7710**	0.7692
Sonar	**0.8487**	0.7694	0.8307	0.7948	0.7726	0.7729	0.7731
Tae	**0.6715**	0.6709	0.4113	0.5553	0.5504	0.5175	0.5461
Thyroid	**0.9704**	0.9256	0.9389	0.9351	0.9332	0.9380	0.9424
Tic-tac-toe	0.8549	0.6906	0.7756	0.7665	0.7045	**0.9833**	0.7770
Vehicle	0.7116	0.6572	0.7175	0.6333	**0.7990**	0.7389	0.6745
Vowel	**0.9824**	0.9788	0.9778	0.5542	0.7970	0.6923	0.4431
Wine	0.9731	0.9706	0.9549	0.9548	0.9438	**0.9775**	0.9625
Yeast	**0.5926**	0.5151	0.5317	0.5814	0.5552	0.5714	0.5180
Avg. values	**0.8403**	0.7976	0.7850	0.7755	0.7805	0.7845	0.7881
Avg. ranks	**2.1833**	4.6667	4.6000	4.9000	3.9667	3.6667	4.0167

conjuntos de datos (en nuestro caso hay siete algoritmos y 30 conjuntos de datos) conforme a una F-distribución. Cuando el test de Iman y Davenport indica que existen diferencias significativas, es necesario realizar un test post hoc, en nuestro caso, el test de Bonferroni–Dunn [16] se emplea para encontrar las diferencias significativas entre los algoritmos de la comparativa múltiple. El test asume que los resultados de dos clasificadores es significativamente distinto si difieren en su ranking al menos un cierto valor que se denomina distancia crítica. Finalmente, el test de pares de Wilcoxon [17] se emplea para evaluar múltiples comparaciones para cada par de algoritmos.

V. Resultados

Esta sección presenta los resultados de los experimentos y procede a su validación mediante test estadísticos no paramétricos.

La Tabla II muestra los resultados medios de exactitud para los diferentes conjuntos de datos (filas) y algoritmos (columnas). El algoritmo gravitacional propuesto mejora los resultados de los otros métodos en 14 de los 30 conjuntos de datos y obtiene resultados competitivos en el resto. A continuación, realizaremos algunas apreciaciones para resaltar los resultados de algunos métodos y conjuntos de datos.

Banana es un conjunto de datos artificial (2 clases y 2 atributos) donde las instancias pertenecen a dos grupos con una forma de banana pero contiene ruido. Las máquinas de vector soporte, C-SVM y SMO, fallan en sus predicciones en este conjunto de datos, prediciendo todos los ejemplos de una única clase.

Hayes-Roth es otro conjunto de datos artificial originalmente creado para evaluar el rendimiento de los clasificadores basados en prototipos. Contiene un atributo generado aleatoriamente, que le añade ruido a los datos. En este caso, el clasificador KNN es el que peor se comporta, puesto que no es capaz de filtrar el ruido de este atributo. Por otro lado, ambos modelos gravitatorios evitan este problema mucho mejor que el resto de los métodos, gracias a la ponderación de los pesos de los atributos, proporcionando unos buenos resultados.

El conjunto de datos de glass identification es un problema del mundo real y procede del servicio de

ciencia forense de EEUU, contiene seis tipos diferentes de vidrio que pueden encontrarse en una escena de crimen. Nuestra propuesta gravitatoria mejora los resultados de las máquinas de vector soporte y mejora levemente los resultados del KNN. Resultados similares se han obtenido con el conjunto de datos Sonar (Minas vs Rocas), que contiene 60 atributos reales en el rango [0.0, 1.0].

El estadístico de Iman y Davenport (distribuído conforme a una F-distribución con seis y 174 grados de libertad) reporta un valor de 6.3461 para la exactitud. El test establece un valor de F-distribución igual a 2.1510 para un nivel de significancia de alpha = 0.05. Este valor es menor que el valor crítico reportado de 6.3461. Entonces, el test rechaza la hipótesis nula y por lo tanto se puede decir que existen diferencias significativas entre los resultados de exactitud de los algoritmos. Concretamente, el p-value reportado por el test de Iman y Davenport es de $4{,}667E^{-6}$.

La Figura 3 muestra la aplicación del test de Bonferroni–Dunn a la exactitud con alpha = 0.05, cuya distancia crítica es de 1.4714. Esta gráfica representa los valores de los rankings de los algoritmos. La distancia crítica se representa mediante una barra horizontal gruesa. Los valores que exceden esta línea son algoritmos que obtienen resultados significativamente diferentes que los del algoritmo de control, que en nuestro caso es el modelo gravitatorio propuesto. De esta forma, se evalúan las diferencias de los otros métodos frente a nuestra propuesta. Los algoritmos situados a la derecha del intervalo crítico son métodos significativamente peores que el algoritmo de control. Observando la figura, todos los demás métodos obtienen resultados peores y significativamente diferentes que nuestra propuesta. SMO obtiene los segundos mejores resultados de ranking, pero aun así, se sitúa más allá de la distancia crítica.

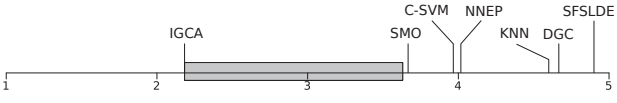


Fig. 3: Test de Bonferroni–Dunn para la exactitud

La Tabla III muestra los resultados del test de Wilcoxon para la exactitud con el objetivo de evaluar múltiples comparaciones entre cada par de algoritmos. SMO es el segundo de los mejores métodos, pero comparado 1 vs 1 con la propuesta obtiene resultados significativamente peores con un p-value de 0.0113. Por otro lado, SFLSDE es el peor de todos los métodos y nunca obtiene resultados de exactitud mejores que nuestra propuesta. El test reporta un p-value en este caso de $1{,}862E^{-9}$.

TABLA III: Test de Wilcoxon para la exactitud

IGCA vs	R^+	R^-	p-value
DGC	419.0	16.0	6.296E-7
KNN	441.0	24.0	1.419E-6
SFLSDE	465.0	0.0	1.862E-9
C-SVM	367.0	98.0	0.004664
SMO	354.0	111.0	0.011304
NNEP	389.0	76.0	7.978E-4

VI. Conclusiones

En este artículo hemos presentado un modelo de clasificación gravitatoria que incluye la ponderación de los pesos de los atributos para cada clase en el cálculo de las distancias entre ejemplos. Los pesos se han optimizado mediante el algoritmo CMA-ES, que ha demostrado encontrar los pesos adecuados de los atributos para cada clase, ignorando atributos ruidosos y potenciando los verdaderamente relevantes. Los efectos de la gravedad alrededor de las instancias ha permitido una correcta clasificación de las instancias, considerando tanto la información global como local de los datos, y proporcionando una buena generalización en zonas alejadas de los ejemplos de entrenamiento. La función del cálculo de la gravedad ha sido adaptada para considerar los conjuntos de datos no balanceados. La propuesta ha logrado mejores resultados de exactitud comparado con los demás algoritmos de clasificación considerados en el estudio experimental, el cual consideraba un clasificación gravitatorio previo. Los resultados han sido validados usando tests no paramétricos por pares y de múltiples comparaciones, cuyos resultados apoyan las diferencias significativamente relevantes entre los resultados de nuestra propuesta y del resto de algoritmos.

Como trabajo futuro se plantea la posibilidad de comprobar el rendimiento del algoritmo frente a conjuntos de datos específicamente no balanceados, y evaluar sus resultados frente a otros algoritmos concretos para este tipo de datos, tanto para clasificación binaria como multiclase. Además también sería posible evaluarlo en conjuntos de datos multietiqueta, donde la clase a predecir no es única. En este sentido, se puede aprovechar la ventaja del algoritmo y tomar no sólo la primera predicción con mayor gravedad, sino las siguientes, pudiendo establecer un orden de preferencia en la predicción de las clases.

Agradecimientos

Este trabajo ha sido financiado por los proyectos del Ministerio de Ciencia y Tecnología y de la Junta de Andalucía, TIN2008-06681-C06-03 y TIC-3720, respectivamente, y los fondos FEDER.

Referencias

[1] M. Paliwal and U. A. Kumar, "Neural Networks and Statistical Techniques: A Review of Applications," *Expert Systems with Applications*, vol. 36, no. 1, pp. 2–17, 2009.

[2] A. Widodo and B. S. Yang, "Support Vector Machine in Machine Condition Monitoring and Fault Diagnosis," *Mechanical Systems and Signal Processing*, vol. 21, no. 6, pp. 2560–2574, 2007.

[3] D. W. Aha, D. Kibler, and M. K. Albert, "Instance-based learning algorithms," *Machine Learning*, vol. 6, no. 1, pp. 37–66, 1991.

[4] T. Cover and P. Hart, "Nearest neighbor pattern classification," *IEEE Transactions on Information Theory*, vol. 13, no. 1, pp. 21–27, 1967.

[5] I. Kononenko and M. Kukar, *Machine Learning and Data Mining: Introduction to Principles and Algorithms*, Horwood Publishing, 2007.

[6] B. Li, Y. W. Chen, and Y. Q. Chen, "The nearest neighbor algorithm of local probability centers," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 38, no. 1, pp. 141–154, 2008.

[7] L. Peng, B. Peng, Y. Chen, and A. Abraham, "Data gravitation based classification," *Information Sciences*, vol. 179, no. 6, pp. 809–819, 2009.

[8] C. Wang and Y. Q. Chen, "Improving nearest neighbor classification with simulated gravitational collapse," *Lecture Notes in Computer Science*, vol. 3612, pp. 845–854, 2005.

[9] Y. Zong-Chang, "A vector gravitational force model for classification," *Pattern Analysis and Applications*, vol. 11, no. 2, pp. 169–177, 2008.

[10] N. Hansen and A. Ostermeier, "Completely derandomized self-adaptation in evolution strategies," *Evolutionary computation*, vol. 9, no. 2, pp. 159–195, 2001.

[11] A. Cano, A. Zafra, and S. Ventura, "Speeding up the evaluation phase of gp classification algorithms on gpus," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, pp. 1–16, 2011.

[12] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, and F. Herrera, "KEEL: A Software Tool to Assess Evolutionary Algorithms for Data Mining Problems," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 13, pp. 307–318, 2009.

[13] J. Alcalá-Fdez, A. Fernandez, J. Luengo, J. Derrac, S. García, L. Sánchez, and F. Herrera, "KEEL Data-Mining Software Tool: Data Set Repository, Integration of Algorithms and Experimental Analysis Framework," *Analysis Framework. Journal of Multiple-Valued Logic and Soft Computing*, vol. 17, pp. 255–287, 2011.

[14] S. García, A. Fernández, J. Luengo, and F. Herrera, "Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power," *Information Sciences*, vol. 180, no. 10, pp. 2044–2064, 2010.

[15] David J. Sheskin, *Handbook of Parametric and Nonparametric Statistical Procedures*, Chapman & Hall/CRC, 2007.

[16] O. J. Dunn, "Multiple Comparisons Among Means," *Journal of the American Statistical Association*, vol. 56, no. 293, pp. 52–64, 1961.

[17] F. Wilcoxon, "Individual Comparisons by Ranking Methods," *Biometrics Bulletin*, vol. 1, no. 6, pp. 80–83, 1945.

[18] J. Giblin, D. Marolf, and R. Garvey, "Spacetime embedding diagrams for spherically symmetric black holes," *General Relativity and Gravitation*, vol. 36, pp. 83–99, 2004.

[19] B. Yang, L. Peng, Y. Chen, H. Liu, and R. Yuan, "A DGC-based data classification method used for abnormal network intrusion detection," *Lecture Notes in Computer Science*, vol. 4234, pp. 209–216, 2006.

[20] H. Liu and L. Yu, "Toward integrating feature selection algorithms for classification and clustering," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 4, pp. 491–502, 2005.

[21] G. V. Lashkia and L. Anthony, "Relevant, irredundant feature selection and noisy example elimination," *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 34, no. 2, pp. 888–897, 2004.

[22] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Transactions on Knowledge and Data Engineering*, vol. 21, no. 9, pp. 1263–1284, 2009.

[23] R. Kohavi, "A Study of Cross-validation and Bootstrap for Accuracy Estimation and Model Selection," in *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, San Francisco, CA, USA, 1995, vol. 2, pp. 1137–1143, Morgan Kaufmann Publishers.

[24] T.S. Wiens, B.C. Dale, M.S. Boyce, and G.P. Kershaw, "Three way k-fold cross-validation of resource selection functions," *Ecological Modelling*, vol. 212, no. 3-4, pp. 244–255, 2008.

[25] G. J. McLachlan, *Discriminant Analysis and Statistical Pattern Recognition*, Wiley-Interscience, New York, 2004.

[26] I. Triguero, S. García, and F. Herrera, "Differential evolution for optimizing the positioning of prototypes in nearest neighbor classification," *Pattern Recognition*, vol. 44, no. 4, pp. 901–916, 2011.

[27] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[28] J. Platt, *Machines using Sequential Minimal Optimization*, MIT Press, Cambridge, MA, USA, 1998.

[29] F. J. Martínez-Estudillo, C. Hervás-Martínez, P. A. Gutiérrez, and A. C. Martínez-Estudillo, "Evolutionary product-unit neural networks classifiers," *Neurocomputing*, vol. 72, no. 1-3, pp. 548–561, 2008.

[30] S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás, "JCLEC: A Java Framework for Evolutionary Computation," *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, vol. 12, pp. 381–392, 2007.

[31] S. García, D. Molina, M. Lozano, and F. Herrera, "A Study on the use of Non-parametric Tests for Analyzing the Evolutionary Algorithms' Behaviour: A Case Study," *Journal of Heuristics*, vol. 15, pp. 617–644, 2009.

[32] J. Demšar, "Statistical Comparisons of Classifiers over Multiple Data Sets," *Machine Learning Research*, vol. 7, pp. 1–30, 2006.